IN THE CLAIMS:

This listing of claims replaces all prior versions and listings of claims in the application.

1. (Currently Amended) A monitoring ~~debug~~ interface, comprising:

logic responsive to a pre-fork event, the pre-fork event responsive to a vfork system ~~fork instruction~~ call, wherein the pre-fork event includes indicia that identifies a child process to be created in accordance with the vfork system ~~fork instruction~~ call.

2. (Original) The interface of claim 1, wherein the indicia comprises a process identifier.

3. (Currently Amended) The interface of claim 1, wherein the pre-fork event is delivered ~~by a parent process~~ to a software monitor.

4. (Original) The interface of claim 3, wherein the parent process was under run-time analysis by the software monitor.

5. (Currently Amended) The interface of claim 3, wherein the pre-fork event is delivered before the vfork system call is executed by the parent process ~~parent process was instrumented for run-time analysis by the software monitor~~.

6. (Currently Amended) The interface of claim 3, wherein the software monitor responds by executing the child process until completion ~~pre-fork event is delivered before the fork instruction call is executed by the parent process~~.

7. (Currently Amended) The interface of claim ~~3~~ 6, wherein the software monitor responds to indicia of completion of the child process by resuming execution of the parent ~~executing the child~~ process until completion.

8. (Currently Amended) The interface of claim 7, wherein the software monitor ensures that the first event pertaining to the parent process and received after completion of the child process is an event denoting completion of a vfork system call ~~responds to indicia of completion of the child process by executing the parent process until completion~~.

9. (Canceled) The interface of claim 8, wherein the software monitor ensures that the first event in the parent process that spawned the child process is the fork event itself.

10. (Currently Amended) A method for controlling the execution of a child process created from a parent process, wherein the parent process is instrumented by a software tool, the method comprising ~~the steps of~~:

receiving indicia that a vfork system call ~~fork instruction~~ will be executed by the parent process;

suspending execution of the parent process;

extracting a process identifier from the indicia of the vfork system call ~~fork instruction~~, the process identifier corresponding to a child process to be generated by the parent process when the parent process executes the vfork system call ~~fork instruction~~;

setting a process monitor thread to observe the child process; and

resuming execution of the parent process to enable the parent process to execute the vfork system call ~~fork instruction~~.

11. (Currently Amended) The method of claim 10, further comprising:

waiting for indicia that the child process has invoked at least one of an exec system call and an _exit system call or has been terminated by an operating system ~~nominally terminated~~; and

setting a process monitor thread to observe the parent process.

12. (Original) The method of claim 11, wherein setting a process monitor thread comprises enabling observation of trace events generated by the parent process.

13. (Currently Amended) The method of claim 10, wherein receiving indicia comprises receiving a pre-fork event.

14. (Original) The method of claim 13, wherein the pre-fork event includes the process identifier.

15. (Original) The method of claim 10, wherein setting a process monitor thread comprises enabling observation of trace events generated by the child process.

16. (Currently Amended) A method for executing a parent process monitored ~~instrumented~~ by a software tool to ensure execution of a child process when the parent process contains a vfork system call ~~fork instruction~~, the method comprising:

determining if a vfork system call ~~fork instruction~~ is about to be executed;

generating a pre-fork event that includes indicia of a child process that will be generated by the vfork system call ~~fork instruction~~;

sending the pre-fork event to the software tool;

waiting for indicia that the software tool successfully processed the pre-fork event;

executing the vfork system call ~~fork instruction~~; and

suspending execution of the parent process.

17. (Currently Amended) The method of claim 16, further comprising:

waiting for indicia that the child process has invoked at least one of an exec system call and an _exit system call or has been terminated by an operating system; and

resuming execution of the parent process.

18. (Original) The method of claim 17, wherein waiting for indicia that the child process has terminated comprises a trace event.

19. (Original) The method of claim 16, wherein indicia of the child process comprises a process identifier.

20-28. (Canceled)

29. (New) A method for controllably switching a target process of a process monitor thread between an instrumented parent process and a child process generated by the parent process, the method comprising:

checking whether the successful initiation of the child process can be asserted;

when the successful initiation of the child process cannot be asserted,

checking if the parent process responsible for creating the child process received indicia of a failure of a vfork system call designated to create the child process;

when the indicia has not been received,

waiting an amount of time before rechecking for the successful initiation of the child process; otherwise,

notifying a software monitor of the unsuccessful initiation of the child process and resuming execution of the parent process;

monitoring the parent process;

otherwise, when the successful initiation of the child process can be asserted,

monitoring the successfully created child process.

30. (New) The method of claim 29, wherein the step of checking whether the successful initiation of the child process can be asserted comprises verifying the success of a trace event by using the process identifier of the child process.

31. (New) The method of claim 29, wherein checking if a parent process responsible for creating the child process received indicia of a failure comprises searching for a trace event while performing a non-blocking trace wait on the parent process.

32. (New) The method of claim 29, further comprising:

aborting child process monitoring when the initiation of the child process is unsuccessful.

33. (New) An operating system, comprising:

a pre-fork event, the pre-fork event responsive to a vfork system call wherein the pre-fork event includes indicia that identifies a child process to be created in accordance with the vfork system call.

34. (New) The operating system of claim 33, wherein the indicia comprises a process identifier.

35. (New) A computer readable medium, comprising:

logic responsive to a pre-fork event, the pre-fork event responsive to a vfork system call wherein the pre-fork event includes indicia that identifies a child process to be created in accordance with the vfork system call.

36. (New) The computer readable medium of claim 35, wherein the indicia comprises a process identifier.